# 一种加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法

周 旭[1] 李肯立[2] 乐光学[1] 朱开乐[1]
(嘉兴学院数理与信息工程学院 嘉兴 314001)[1] (湖南大学信息科学与工程学院 长沙 410082)[2]

**摘 要** 加群 $Z_p^+$ 上离散对数问题在公钥密码系统分析中具有非常广泛的应用。研究一种加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法。算法主要由解空间生成器、并行乘法器、并行加法器、解转换器及解搜索器组成。其中解空间生成器借鉴传统计算机中3表算法的思想,将解空间的生成分为3个部分来完成,极大减少了非法解的搜索空间。本算法的生物操作时间复杂度为 $O(k^2)$,需要 $O(1)$ 个试管数、$O(2^k)$ 条 DNA 链,最长 DNA 链长为 $O(k^2)$(其中 $k$ 为加群上离散对数问题群阶 $p$ 的二进制编码位数)。最后,通过 DNA 计算通用的试验方法对算法进行了仿真,验证了算法的可行性和有效性。

**关键词** DNA 计算,NP 完全问题,密码分析,加群 $Z_p^+$ 离散对数问题
**中图法分类号** TP306.1 **文献标识码** A

## Fast Parallel Molecular Algorithm for Solving the Discrete Logarithm Problem over Group on $Z_p^+$ DNA-based Computing

ZHOU Xu[1] LI Ken-li[2] YUE Guang-xue[1] ZHU Kai-le[1]
(College of Mathematics and Information Engineering, Jiaxing University, Jiaxing 314001, China)[1]
(School of Computer and Communications, Hunan University, Changsha 410082, China)[2]

**Abstract** The discrete logarithm problem over group $Z_p^+$ is widely applied in the public key cryptosystems. We proposed a new DNA computing algorithm to solve the problem. Our new algorithm consists of an initial solution generator, a parallel multiplier, an invalid parallel detector, a parallel conventor and a parallel solution searcher. For the sake of reducing the DNA sequences required in the new algorithm, the solution space will be generated considering the three-list algorithm. The proposed algorithm needs $O(k^2)$ biological operations, $O(1)$ test tubes, $O(2^k)$ DNA sequences, and the maximum length of the DNA sequence is $O(k^2)$. Finally, the common test methods were used to verify the new algorithm's feasibility and effectiveness.

**Keywords** DNA-based computing, NP-complete problem, Cryptoanalysis, Discrete logarithm problem ever group $Z_p^+$

## 1 引言

自从 DNA 计算模型问世以来,其由于具有海量存储能力和巨大并行,逐渐引起了人们的关注[1-10]。现今 DNA 计算逐渐成为计算机科学和生物学领域的一个新的研究热点,并且在密码分析领域、科学计算、医疗等方面展现出了巨大的潜力[1-9]。

现今 DNA 计算不仅能作为解决 NP 问题的一种潜在方案,并且为密码分析领域提供了一种新的兼具理论和实验可行性的思路[5-10]。文献[5]提出了一种破解 DES 的生物计算算法,文献[6]设计了一种 Pollard $p-1$ 因子分解的 DNA 计算机改进算法,文献[7]设计了一种基于 DNA 计算自组装模型的 Diffie-Hellman 算法破译,文献[8]设计了一种基于 DNA 计算自组装模型的 NTRU 密码系统破译方案,文献[9]提出一种基于 Adleman-Lipton 模型的多项式时间内 RSA 密码系统的破译方案,文献[10]提出了一种基于 Adleman-Lipton 模型的椭圆曲线的 DNA 计算算法。本文则主要针对加群 $Z_p^+$ 上离散对数问题(DLP),提出一种能够在多项式时间内解决加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法。

本文第2节介绍相关的理论背景,包括加群上离散对数问题描述以及本文所用到的改进 DNA 计算模型;第3节提出求解加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法及相关子算法;第4节给出实验模拟;最后给出本文的结论与下一步工作目标。

## 2 相关理论背景

### 2.1 加群 $Z_p^+$ 上离散对数问题

设群 $G=(Z_p,+)$,$\alpha,\beta\in G$,且 $p$ 为群 $G$ 的阶。求整数 $l(0\leqslant l\leqslant p-1)$ 满足 $\beta=(l\times\alpha) \bmod p$,即加群 $G$ 上的离散对数问题。由于离散对数的难解性,使得它被广泛应用于密码学

到稿日期:2011-05-17 返修日期:2011-09-29 本文受国家自然科学基金(60603053,90715029),教育部新世纪优秀人才支持计划(NCET-08-0177),浙江省自然科学基金(Y1090264),浙江省大学生新苗计划项目(851910123),嘉兴市科技计划项目(2011AY1003),浙江省公益性技术应用研究计划项目(2011C23130),嘉兴学院科研校内重点课题(70110X01BL)资助。

周 旭(1983-),女,助教,主要研究方向为并行计算、DNA 计算,E-mail:happypanda2006@126.com;李肯立(1971-),男,教授,CCF 高级会员,主要研究方向为并行处理、生物计算、网格计算等。

中[11]。

若 $p$ 值较小，该问题可以通过穷举搜索来解决。假设 $p=11,\alpha=3,\beta=7$，求解满足 $7=3\times l \bmod 11$ 的 $l$ 值。使用穷举搜索的方法，将搜索 $p$ 个密钥空间，即 $l$ 取值从 0～10 尝试所有可能值，直到找到一个满足该等式的结果：

$l=0,3\times 0 \bmod 11=0$；$l=1,3\times 1 \bmod 11=3$；

$l=2,3\times 2 \bmod 11=6$；$l=3,3\times 3 \bmod 11=9$；

$l=4,3\times 4 \bmod 11=1$；$l=5,3\times 5 \bmod 11=4$；

$l=6,3\times 6 \bmod 11=7$；$l=7,3\times 7 \bmod 11=10$；

$l=8,3\times 8 \bmod 11=2$；$l=9,3\times 9 \bmod 11=6$；

$l=10,3\times 10 \bmod 11=8$。

通过以上穷举搜索得到此离散对数问题的答案之一为 $l=6$。对于 $p$ 值比较小的情况，通过穷举查找法能够很容易求解该问题。但对于 $p$ 值比较大的情况下，假设是一个可以编码为 60 位二进制的素数，那么最坏情况下需要搜索 260 个密钥空间，以现有的最快的电子计算机亦无法在有效的时间内求解该问题[12,13]。

### 2.2 一种改进的 DNA 计算模型

算法基于原有 Adleman-Lipton 模型的生物操作[9,13]，并进行改善，引入了如下生物操作：

(1)$Cut(T,x_i)$：给定试管 $T$，此操作将试管中 DNA 链含有的子链 $x_i$ 切除；

(2)$AppendHead(T,x,z)$：给定试管 $T$ 和一个短的 DNA 链 $x,z$，首先找到短链 $x$ 所在的位置，将链 $z$ 添加到 $T$ 中每个链 $x$ 前端；

(3)$AppendTail(T,z)$：给定试管 $T$ 和一个短的 DNA 链 $z$，该操作将链 $z$ 添加到 $T$ 中每个链的末尾。

## 3 加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法

### 3.1 加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法思想

本文基于加群 $Z_p^+$ 上离散对数问题的特性及 DNA 计算的优点，提出了一种新的加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法。算法中为了减少实验时间及搜索空间，借鉴了传统计算机中 3 表算法的思想，将解空间的生成分为 3 部分来完成。

**算法 1** 求解 $Z_p^+$ 加法群上 DNA 计算算法框架

1)首先得到离散对数问题的初解空间 $T_0$，$T_0$ 中含有 $\alpha$、$2\alpha$、$3\alpha$ 编码后得到的 DNA 链，其中编码分为两部分，即系数编码和值编码。

2)检查 $T_0$ 中是否有等于 $\beta$ 的 DNA 链：若有，则找到系数 1，否则执行步骤 3)。

3)重复以下步骤，直到遍历群上所有元素，从而找到 1。

(1)分别复制 $T_0$ 中的 DNA 链到试管 $T_1$、$T_2$ 与 $T_3$ 中，对试管 $T_2$ 中的 DNA 链进行并行加 $3^i\alpha$ 操作，试管 $T_3$ 进行并行加 $2*3^i\alpha$ 操作。

(2)检测试管 $T_2$ 中的链是否大于等于 $p$，得到两个试管 $T_2^{\geqslant}$ 与 $T_2^{<}$。

(3)若试管 $T_2^{\geqslant}$ 中非空，则试管 $T_3$ 中包含的 DNA 链的编码值全部大于 $p$。将试管 $T^{\geqslant}$ 与 $T_3$ 的 DNA 链合并到试管 $T_2^{\geqslant}$ 中。对试管 $T^{\geqslant}$ 中的 DNA 链进行合法化处理。最后将试管 $T_1$ 与 $T_2^{\geqslant}$ 与 $T_2^{<}$ 中的 DNA 链合并到试管 $T_1$ 中。

(4)若检测试管 $T_2^{\geqslant}$ 中为空，则根据试管 $T_3$ 中的链是否大于等于 $p$，得到两个试管 $T_3^{\geqslant}$ 与 $T_3^{<}$。若检测试管 $T_3^{\geqslant}$ 中非空，则将试管 $T_3^{\geqslant}$ 中的 DNA 链进行合法化处理。最后再将试管 $T_1$、$T_2^{<}$、$T_3^{\geqslant}$ 与 $T_3^{<}$ 中的 DNA 链合并到试管 $T_1$ 中。

(5)检查 $T_1$ 中是否有等于 $\beta$ 的 DNA 链：若有，则找到系数 l，终止循环。

从以上解决加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法思想中发现，要实现该思想，必须设计关于群 $G$ 的并行加法器、并行乘法器、初始解空间生成器、对检测结果进行检测的并行检测器、将非法值转换为合法值的并行合法化处理器以及解搜索器。以上相应子算法将在 3.2 节－3.5 节中介绍。

### 3.2 并行初始解空间生成器

DNA 计算中解空间的生成主要借助于生物分子学中的复制和添加操作。本文的新算法中，解空间的生成及完善主要基于加法运算和乘法运算。以下将针对以上两种运算分别设计相应的并行加法器、并行乘法器。

#### 3.2.1 *并行加法器*[9]

鉴于上述算法思想，本加法器在一次并行加法过程中所有 DNA 链只需加群 $G$ 上的同一个值。下列算法中，$M$ 表示加数。为了便于进行加法运算，将 $M$ 编码为 $m_1m_2\cdots m_k$，$x_1x_2\cdots x_k$ 表示链上被加数的编码，$y_1y_2\cdots y_{k+1}$ 表示加法结果，其中 $m_1,x_1,y_1$ 分别代表最底位，$m_k,x_k,y_k$ 代表最高位。定义标志符 $m_i^0$ 表示 $m_i$ 的值为'0'，$m_i^1$ 表示 $m_i$ 的值为'1'；$x_i^0$ 表示 $x_i$ 的值为'0'，$x_i^1$ 表示 $x_i$ 的值为'1'；$y_j^0$ 表示 $y_j$ 的值为'0'，$y_j^1$ 表示 $y_j$ 的值为'1'(其中 $1\leqslant i\leqslant k,1\leqslant j\leqslant k+1$)。

并行加法器的具体实现过程参见文献[9]。此并行加法器算法 ParallelAdder($T_{01},T_{02},m_1\cdots m_k$)为加法群 $Z_p^+$ 上的并行加法器。算法共使用了 $O(k)$ 个抽取操作、$O(k)$ 个合并操作、$O(k)$ 个尾部添加操作以及 8 个试管。

#### 3.2.2 *并行乘法器*

本文的 DNA 计算算法中需要多次使用乘 2 及乘 3 运算，为此特设计了以下分别进行乘 2 运算及乘 3 运算的并行乘法器。基本思想：如式(1)所示，乘 2 运算转换成二进制移位运算，首先将 $m_1\cdots m_k$ 中 $k$ 位二进制数全部向前移动一位，移位操作完成后在最后的空位添加一个 0。可以通过基本生物操作 $AppendHead(T_0,m_1,m_0^0)$ 来执行乘 2 运算。首先找到二进制的最低位 $m_1$，在最低位之前加入一位 $m_0^0$。

$$m_1\cdots m_k\times 2=m_0^0m_1\cdots m_k \tag{1}$$

乘 3 运算见式(2)，可以通过乘 2 运算及加法运算得到最后结果。

$$m_1\cdots m_k\times 3=m_1\cdots m_k\times 2+m_1\cdots m_k \tag{2}$$

**算法 2** 乘 3 并行乘法器

Procedure ParallelMultiplierThree($T_0,m_1\cdots m_k,3$)

(1)AppendHead($T_0,m_1,m_0^0$)

(2)ParallelAdder($T_0,m_1\cdots m_k$)

EndProcedure

ParallelMultiplierThree($T_0,m_1,\cdots,m_k,3$)算法共使用了 $O(k)$ 个抽取操作、$O(k)$ 个合并操作、$O(k)$ 个尾部添加操作、1 个头部添加操作以及 6 个试管。

#### 3.2.3 *初始解空间生成器*

为了解决加群 $Z_p^+$ 上离散对数问题，首先设计了如下初始解空间生成器。生成含有 $\alpha$、$2\alpha$、$3\alpha$ 系数及相应值编码后得到的 DNA 链。MakeValue($\alpha$)用于对 $\alpha$ 进行二进制编码，编码为 $k$ 位二进制的形式。

**算法 3** 初始解空间生成器

Procedure InitialSolutionGenerator($T_0,\alpha$)

1) MakeValue($\alpha$)

2) For j=1 down to k
3) If($\alpha_j$==1) Then
4) AppendTail($T_0$,$\alpha_j^1$)
5) Else
6) AppendTail($T_0$,$\alpha_j^0$)
7) EndIf
8) EndFor
9) Amplify($T_0$,$T_1$,$T_2$,$T_3$)
10) Cut($T_2$,$\alpha_k^0$)
11) AppendHead($T_1$,$\alpha_1$,$c_1^1c_2^0c_3^0\cdots c_k^0$)
12) AppendHead($T_2$,$\alpha_1$,$\alpha_0^0$)
13) AppendHead($T_2$,$\alpha_0$,$c_1^0c_2^1$ $c_3^0\cdots c_k^0$)
14) ParallelMultiplierThree($T_3$,$\alpha_1$,$\cdots$,$\alpha_k$,3)
15) AppendHead($T_3$,$\alpha_1$,$c_1^1$ $c_2^0$ $c_3^0\cdots c_k^0$)
16) 将试管中的DNA链的标号进行重新编排,即代表 α、2α、3α 的 k 位二进制重新编排为 $\alpha_1^0\alpha_2^1\cdots\alpha_k^0$
17) $T_0$=Merge( $T_0$,$T_1$,$T_2$,$T_3$)
EndProcedure

算法 InitialSolutionGenerator($T_0$,α)用于生成初始解空间。它共使用了 O($k$)个尾部添加操作、O($k$)个合并操作、2个头部添加操作、1个复制操作以及 6 个试管。对于每一位的二进制数分别产生长度为 15 的 DNA 序列来标识,则初始解空间生成后 DNA 链长为 $15\times k\times 2=30k$,DNA 链数为 3。

### 3.3 并行检测器

根据加群 $Z_p^+$ 上离散对数问题的定义,在计算过程中若出现最后结果大于 $p$ 的解链,则必须进行合法化处理。为了搜索出非法解链,设计了并行检测器。将运算结果 $y$ 编码为 $y_1\ y_2\cdots y_{k+1}$,$p$ 编码为 $p_kp_{k-1}\cdots p_1$,通过执行并行检测器的操作,得到两个试管 $T^{\geqslant}$ 及 $T^{<}$。其中试管 $T^{\geqslant}$ 含有大于等于 $p$ 的非法解链,试管 $T^{<}$ 含有小于 $p$ 的合法值的解链。为了使得本算法更加具有实验可行性,能够减少 DNA 链长及实验过程中的错误率,以下算法将 $p$ 值编码的 DNA 链放置于独立的试管 $T_p$ 中,而并非通常地将 $p$ 值的编码添加到试管 $T_0$ 中所有的 DNA 链后。

**算法 4** 并行非法 DNA 链检测器

Procedure ParallelDetector-Invalid($T_0$,p,$T^{\geqslant}$,$T^{<}$)
1) For j=k down to 1
2) If($p_j$==1) Then
3) AppendTail($T_p$,$p_j^1$)
4) Else
5) AppendTail($T_p$,$p_j^0$)
6) EndIf
7) EndFor
8) ($T_1$,$T_2$)=Extract($T_0$,$y_{k+1}$)
9) $T^{\geqslant}$=Merge($T^{\geqslant}$,$T_1$)
10) For i=k to 1
11) ($T_1$,$T_2$)=Extract($T_0$,$p_i$)
12) If (Detect($T_0^0$)='yes') Then
13) ($T^{\geqslant}$,$T_0$)=Extract($T_0$,$y_i$)
14) Else
15) ($T_0$,$T^{<}$)=Extract($T_0$,$y_i$)
16) EndIf
17) EndFor
EndProcedure

算法 ParallelDetector-Invalid($T_0$,$T_p$,$T^{\geqslant}$,$T^{<}$)为群 $Z_p^+$ 上并行非法 DNA 链检测器。在本算法中,共使用了 O($k$)个抽取操作、1 个合并操作、O($k$)个尾部添加操作以及 6 个试管。此外,此算法用于搜索出所有的非法解链,算法执行前后 DNA 链数及链长将不会发生改变。

### 3.4 并行转换器

加群离散对数问题中对于检测出来的非法值,即大于阶数 $p$ 的值需要进行合法化处理,即对这些值进行并行减 $p$ 运算。假设并行合法化后最终得到的结果为 $d$,相应的编码为 $d_1\cdots d_k$。并行转化器中大多数情况需要多次减 $p$ 运算,因而为了减少 DNA 链长,特引入 DNA 缓存概念,将转化器中间运算结果存储到相应 DNA 缓存中。

**算法 5** 并行转换器

Procedure ParallelConvertor($T_0$,$T_p$)
1) $T_Y$=∅,$T_N$= $T_0$
2) For i=1 to k
3) ($T_1$,$T_2$)=Extract($T_N$,$y_i$) and( $T_3$,$T_4$)=Extract($T_Y$,$y_i$)
4) $T_2$=$\bigcup$($T_2$,$T_3$)
5) If(Detect($T_p$,$p_i^0$)='yes') Then
6) AppendTail($T_1$,$d_i^1$) and AppendTail($T_2$,$d_i^0$)
7) AppendTail($T_4$,$d_i^1$)
8) $T_N$=Merge($T_1$,$T_2$) and $T_Y$=Merge($T_Y$,$T_Y$,$T_4$)
9) Else
10) AppendTail($T_1$,$d_i^0$) and AppendTail($T_2$,$d_i^1$)
11) AppendTail($T_4$,$d_i^0$)
12) $T_N$=Merge($T_N$,$T_N$,$T_1$) and $T_Y$=Merge($T_2$,$T_2$,$T_4$)
13) EndIf
14) EndFor
15) $T_0$=Merge($T_Y$,$T_N$)
16) AppendTail($T_0$,$d_{k+1}^0$)
17) $T_0$=Merge($T^{\geqslant}$,$T^{<}$)
18) ParallelDetector-Invalid($T_0$,p,$T^{\geqslant}$,$T^{<}$)
19) If(Detect($T^{\geqslant}$)== 'yes') Then
20) ParallelConvertor($T_0$,$T_p$)
21) EndIf
EndProcedure

算法 ParallelSubtractor($T_0$,$T_p$)为群 $Z_p^+$ 上的并行转换器,实现解空间的合法化处理。算法共使用了 O($k$)个抽取操作、O($k$)个合并操作、O($k$)个添加操作以及 8 个试管。算法将对所有值大于 $p$ 的 DNA 链进行合法化处理。

### 3.5 并行解搜索器

为了搜索出加法群上离散对数问题的最后解链,设计了如下解链搜索器。

**算法 6** 并行解搜索器

Procedure ParallelSolutionSearcher($T_0$,β,$T^{=}$,$T^{\neq}$)
1) For j=k down to 1
2) If($\beta_j$==1) Then
3) AppendTail($T_{answer}$,$\beta_j^1$)
4) Else
5) AppendTail($T_{answer}$,$\beta_j^0$)
6) EndIf
7) EndFor
8) For i=k to 1
9) ($T_1$,$T_2$)=Extract($T_0$,$\beta_j$)
If(Detect($T_1$)='yes') Then
10) ($T^{=}$,$T^{\neq}$)=Extract($T_0$,$y_i$)
11) EndIf
12) If(Detect($T_2$)='yes') Then
13) ($T^{\neq}$,$T^{=}$)=Extract($T_0$,$y_i$)
EndIf

14) EndFor
15) EndProcedure

算法 ParallelSubtractor($T_0$, $T_p$)为群 $Z_p^+$ 上的并行减法器。算法共使用了 O($k$)个抽取操作、O($k$)个合并操作、O($k$)个添加操作以及 5 个试管。

### 3.6 加群 $Z_p^+$ 上离散对数问题 DNA 计算算法

鉴于 3.1 节中的解决群 $Z_p^+$ 上离散对数问题 DNA 计算算法思想，以及上述子算法，设计解决群 $Z+_p$ 上离散对数问题的算法如下：

**算法 7** 解决群 $Z_p^+$ 上离散对数问题 DNA 计算算法

Procedure DLPAlgorithm($T_0$, p)
1) InitialSolutionGenerator($T_0$, α)
2) ParallelDetector($T_0$, β, $T^=$, $T^{\neq}$)
3) If(Detect($T^=$)== 'yes') Then
4) Read($T^=$)
5) Else
6) $T_0$=Merge($T_0$, $T_0$, $T^{\neq}$)
7) For i=1 to $\log_3 \frac{p-1}{2}$
8) Amplify($T_0$, $T_1$, $T_2$, $T_3$)
9) ParallelAdder($T_2$, $c_1$, …, $c_k$, $3^i$) and ParallelAdder($T_2$, $x_1 \cdots x_k$, $3^i\alpha$)
10) ParallelAdder($T_3$, $c_1$, …, $c_k$, $2*3^i$) and ParallelAdder($T_3$, $x_1 \cdots x_k$, $2*3^i\alpha$)
11) ParallelDetector-Invalid($T_2$, p, $T_2^{\geqslant}$, $T_2^{\leqslant}$)
12) If(Detect($T_2^{\geqslant}$)== 'yes') Then
13) $T_2^{\geqslant}$=Merge($T_2^{\geqslant}$, $T_2^{\geqslant}$, $T_3$)
14) ParalleSubtractor($T_2^{\geqslant}$, $T_p$)
15) $T_1$=Merge($T_1$, $T_2^{\geqslant}$, $T_2^{\leqslant}$)
16) Else
17) ParallelDetector-Invalid($T_3$, p, $T_3^{\geqslant}$, $T_3^{\leqslant}$)
18) If(Detect($T_3^{\geqslant}$)== 'yes') Then
19) ParalleSubtractor($T_3^{\geqslant}$, $T_p$)
20) $T_1$=Merge($T_1$, $T_1$, $T_2^{\leqslant}$, $T_3^{\geqslant}$, $T_3^{\leqslant}$)
21) EndIf
22) EndIf
23) ParallelDetector($T_1$, β, $T^=$, $T^{\neq}$)
24) If(Detect($T^=$)== 'yes') Then
25) Read($T^=$)
26) Break;
27) Else
28) $T_0$=Merge($T_0$, $T^{\neq}$)
29) EndIf
30) EndFor
31) EndProcedure

**引理 1** 算法 DLPAlgorithm($T_0$, $p$)可以解决群 $Z_p^+$ 上的离散对数问题。

证明：步骤(1)首先初始化解空间，得到关于 $\alpha$、$2\alpha$、$3\alpha$ 的 DNA 编码。步骤(2)使用解搜索器在合法 DNA 链中找出运算结果等于 $\beta$ 的解链。步骤(3)检测试管 $T^=$，若其非空，则找到加法群上离散对数问题的解，读出解链上的结果信息，否则执行步骤(6)－(30)，将试管 $T_0$，$T^{\neq}$ 中的 DNA 链合并至试管 $T_0$ 中。步骤(7)－(30)为一循环过程。其中，步骤(8)将 $T_0$ 复制到 $T_1$ 和 $T_2$ 与试管 $T_3$ 中；步骤(9)调用算法将 $T_3$ 中每条链上当前系数值加 $3^i$，将 $T_2$ 中每条链所有 DNA 链上当前值加 $3^i\alpha$；步骤(10)调用算法将 $T_3$ 中每条链上当前系数值加 $2*3^i$，将 $T_3$ 中每条链上当前值加 $2*3^i\alpha$；步骤(11)检测 $T_2$ 中加法结果的非法值，将非法值保存在试管 $T_2^{\geqslant}$ 中，合法值保存在试管 $T_2^{\leqslant}$ 中；步骤(12)首先检测试管 $T_2^{\geqslant}$，若试管 $T_2^{\geqslant}$ 中非空，则执行步骤(13)－(20)，将运算的中间非法结果进行合法化处理；步骤(23)使用解搜索器在合法 DNA 链中找出运算结果等于 $\beta$ 的解链。若试管 $T^=$ 非空，则找到加法群上离散对数问题的解，读出解链上的结果信息，并退出循环；否则将试管 $T_0$，$T^{\neq}$ 中的 DNA 链合并至试管 $T_0$ 中。

### 3.7 性能分析

本文提出的求解加群 $Z_p^+$ 上离散对数问题的 DNA 计算算法分为两大部分：首先通过初始解空间生成器生成问题的初始解空间。其次步骤(2)为一并行解搜索算法。步骤(3)执行 1 次检测操作，若检测试管非空，则执行步骤(4)，否则将执行步骤(6)－(30)。步骤(7)－(30)为一循环过程，循环的次数由群阶数 $p$ 决定。综合以上两部分，本文算法的实现需要执行 O($k^2$)个抽取操作、O($k^2$)个添加操作、O($k^2$)个合并操作、O($k$)个复制操作以及 O(1)个试管、O($2^k$)条 DNA 链，最长 DNA 链长为 O($k^2$)，其中变量 $k$ 表示 $q$ 二进制编码的位数。

此外，与传统解决 NP 等困难问题的 DNA 计算算法相比，本文算法借鉴了传统计算机算法中经典的 3 表算法，即将解空间的生成分为 3 个部分来完成，这样不仅能通过搜索进一步减少解空间，而且能减少了生物操作，进而减少错误率，增加算法实现的可行性。

## 4 实验模拟

为验证本文算法的有效性，以 $p=11$，$\alpha=3$，$\beta=7$ 作为实例，求解满足 $7=3\times l \bmod 11$ 的 $l$ 值。为了描述方便，特定义如下变量：$c_i$ 代表系数 $l$ 对应的二进制编码；$x_i$ 代表 $3\times l$ 值对应的二进制编码；$y_j$ 是解空间变换过程中的中间运算结果对应的二进制编码；$d_j$ 是进行合法化的中间运算结果用标示。其中 $1\leqslant i\leqslant 4$，$1\leqslant j\leqslant 5$。

本文借鉴了文献[9]中使用的计算模型[10]，每一个信息元对应两个长度为 15 的碱基"值序列"。依据 Braich 编码规则，在 Windows XP 操作系统下使用 Visual C++6.0 的编译器来产生 DNA 序列，如表 1 所列。

表 1 试管 $T_0$ 中各位的 DNA 序列

| 位 | 5′→3′DNA 序列 | 位 | 5′→3′DNA 序列 |
|---|---|---|---|
| $c_1^0$ | TAATCTTCTTCCTCA | $c_1^1$ | CTTACTTACCAACAT |
| $c_2^0$ | TTATACACTATCTTC | $c_2^1$ | CCACCAAACCTAAAC |
| $c_3^0$ | TCTCACTCTAATCAT | $c_3^1$ | CAACTAATACTATAA |
| $c_4^0$ | ACTCACATACACCAC | $c_4^1$ | CAACCTATTAACTTA |
| $x_1^0$ | ATCTAAACCATATAA | $x_1^1$ | TTTAATTCATCCTAA |
| $x_2^0$ | AAATCTCACTACATA | $x_2^1$ | AAATATACCAACTCC |
| $x_3^0$ | CTCTTTCCCATCATC | $x_3^1$ | ATCTATTCCTTTATT |
| $x_4^0$ | CTCTCTCAAATTCAC | $x_4^1$ | CCTTATCTCTTAAAT |
| $y_1^0$ | CCCTACAACTCACAT | $y_1^1$ | ATCCTCCACTTCCAA |
| $y_2^0$ | TCACTCTTTAATCCT | $y_2^1$ | AACATAATTATTCCT |
| $y_3^0$ | ATTTACTCATCCTAC | $y_3^1$ | CCTCCTTAACACCTT |
| $y_4^0$ | CTAATACTTCCATAA | $y_4^1$ | TCTCATTCATCCACA |
| $y_5^0$ | TACCTATATCCACTA | $y_5^1$ | ACACTTAACACCATT |
| $d_1^0$ | CTCAACCAATTCATC | $d_1^1$ | ACAACTACCAATACT |
| $d_2^0$ | TACAACTCCACACAT | $d_2^1$ | TTCCAACCACACTTA |
| $d_3^0$ | CTAATACATTCAACA | $d_3^1$ | CTAATCCTAATAACC |
| $d_4^0$ | CTCTACCTCCTCTTC | $d_4^1$ | CAACCCTCTCTCCAC |
| $d_5^0$ | AACAATTACAACACT | $d_5^1$ | CTCTAACTCCATCAC |

(下转第 268 页)

表1 时间和内存使用性能统计

| 并行线程数 | 帧频 (fps) | 平均处理时间 (ms) | 内存使用情况 (Mb) | 理论帧间隔 (fps) |
|---|---|---|---|---|
| 2 | 10 | 23.0 | 18.4 | 100 |
| | 15 | 23.2 | 18.9 | 66 |
| | 30 | 22.7 | 19.7 | 33 |
| 10 | 10 | 27.0 | 32.1 | 100 |
| | 15 | 28.2 | 32.8 | 66 |
| | 30 | 29.7 | 33.1 | 33 |
| 20 | 10 | 30.3 | 44.7 | 100 |
| | 15 | 31.8 | 45.8 | 66 |
| | 30 | 33.5 | 41.3 | 33 |
| 40 | 10 | 38.2 | 72.4 | 100 |
| | 15 | 34.7 | 73.1 | 66 |
| | 30 | 40.1 | 73.3 | 33 |
| 80 | 10 | 46.7 | 118 | 100 |
| | 15 | 41.3 | 119.3 | 66 |
| | 30 | 44.2 | 119.9 | 33 |

仿真实验结果表明，在并发处理2路和10线程时，单帧处理时间远小于理论计算得到的帧间隔时间，即达到实时处理的效果；在并发处理20路、40路或80路线程时，由于程序运行在单一服务器上，处理时间需要考虑操作系统基本的线程轮询的时间消耗，在视频的帧率为30fps时，处理时间比理论帧间隔时间长，例如，在并发线程数为40时，30fps实际平均处理时间为40.1ms，而理论值为33ms，实际值比理论值大，但是实际图像处理效果仍然良好，视频播放同样流畅，能达到实际工程应用要求。

**结束语** 本文提出的在并发多线程环境下对微动视频目标进行检测和提取的算法，结合了K均值聚类算法和肤色检测算法。根据K均值聚类算法的性质，它具有严密性和运算简单的特征；根据固定肤色检测算法的性质，它具有定位肤色区域快捷高效的特征。在实时处理要求很高的情况下，运用多线程思想对实际应用中的多路访问情况进行解决并实行，从时间性能和准确性能上均具有很好的效果。

## 参考文献


[1] Jasper S, Babak T, Yulia E, et al. Automatic segmentation of video to aid the study of faucet usability for older adults[C]// IEEE Computer Society Conference, Computer Vision and Pattern Recognition Workshops. 2010:63-70

[2] 李实，刘乃琦，郭建东. 多核架构下的多线程负载平衡[J]. 计算机应用，2008，28(12)：139-140

[3] 刘权胜，杨洪斌，吴悦. 同时多线程技术[J]. 计算机工程与设计，2008，29(4)：963-967

[4] Sun Zhi-hai, Zhu Shan-an, Zhang Da-wei. Real-time and Automatic Segmentation Technique for Multiple Moving Objects in Video Sequence[C]// IEEE International Conference, Control and Automation. 2007:825-829

[5] Qu Zhong, Ma Qing-wei. An Algorithm of Slight Moving Object Extraction in Real-time Video[J]. Applied Mechanics and Materials, 2011, 63-64:603-606

[6] Zhan Chao-hui, Duan Xiao-hui, Xu Shuo-yu, et al. An Improved Moving Object Detection Algorithm Based on Frame Difference and Edge Detection[C]// Fourth International Conference on Image and Graphics. 2007:519-523

[7] Qu Zhi-guo, Wang Ping, Gao Ying-hui, et al. Contour detection based on contextual influences[C]// IEEE International Conference on Information and Automation. 2010:1694-1699

[8] Guan Yu-dong, Wang Ying, Zhang Hong. Video object segmentation algorithm integrating temporal and spatial information[C]// 9th International Conference, Electronic Measurement & Instruments. 2009, 4:136-140

[9] 徐保勇，王媛丽，王平，等. 基于高斯混合模型机载下视运动目标检测方法[J]. 重庆理工大学学报：自然科学版，2011，25(11)：56-62


(上接第235页)

**结束语** 本文提出了一种求解加法群 $Z_p^+$ 上离散对数问题的DNA计算算法及相关子算法。解空间的生成算法借鉴传统计算机中经典的3表算法思想，其搜索空间小，减少了实验过程中的错误率，进一步提高了算法的实验可行性。从文献[5-11]及本文算法可以发现，DNA计算为密码分析学提供了一种更加有效的可行方案。当未来DNA计算的生物技术进一步成熟后，现存的密码系统的安全性将面临很大的挑战。

## 参考文献


[1] Harif Ullah A M M. A DNA-based computing method for solving control chart pattern recognition problems[J]. CIRP Journal of Manufacturing Science and Technology, 2011, 133:1-11

[2] Chen R C S, Yang S J H. Applying DNA computation to intractable problems in social network analysis[J]. BioSystems, 2010, 101:222-232

[3] Shin S Y, Lee I H, Kim D M, et al. Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing[J]. IEEE Transactions on Evolutionary Computation, 2005, 9(2): 143-158

[4] 李人厚，余文. 关于DNA计算的基本原理与探讨[J]. 计算机学报，2001，24(9)：973-978

[5] Boneh D, Dunworth C, Lipton R J. Breaking DES Using a Molecular Computer[C]// Proceedings of DIMACS Workshops on DNA Computing. Princeton: American Mathematical Society, 1995:37-65

[6] 王静，李肯立，许进. Pollard $p-1$ 因子分解的DNA计算机改进算法[J]. 系统仿真学报，2008，20(18)：4835-4839

[7] 陈智华. 基于DNA计算自组装模型的Diffie-Hellman算法破译[J]. 计算机学报，2008，31(12)：2116-2122

[8] 张勋才，牛莹，崔光照，等. 基于自组装DNA计算的NTRU密码系统破译方案[J]. 计算机学报，2008，31(12)：2129-2137

[9] Chang W L, Guo Min-yi, Ho Shan-hui, et al. Fast Parallel Molecular Algorithm for DNA-based Computation: Factoring Integers[J]. IEEE Transaction on Nanobioscience, 2005, 4(2): 149-163

[10] Li Ken-li, Zou Shu-ting, Xv Jin. Fast Parallel Molecular Algorithms for DNA-based Computation: Solving the Elliptic Curve Discrete Logarithm Problem over GF(2^n)[J]. Journal of Biomedicine and Biotechnology, 2008(5): 749-752

[11] 章照止. 现代密码学基础[M]. 北京：北京邮电大学出版社，2004，4：152-171

[12] 周康，同小军，许进. 背包问题的闭环DNA算法[J]. 系统仿真学报，2008，20(17)：4605-4608

[13] 周旭，李肯立，乐光学，等. 一种基于分治的三维匹配问题DNA计算算法[J]. 电子学报，2010，38(8)：1831-1836

[14] 卢圣栋. 现代分子生物学实验技术[M]. 北京：中国协和医科大学出版社，1999：23-59